*Laboratoire 1.2 – Introduction à Git*

**Remise** : Déposez ce document sur Moodle.

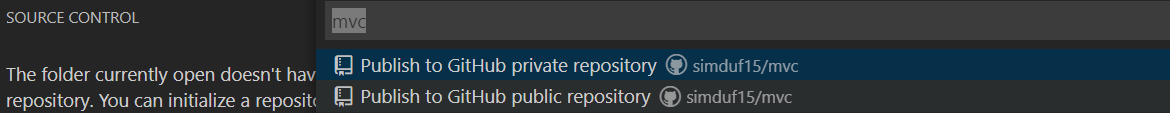
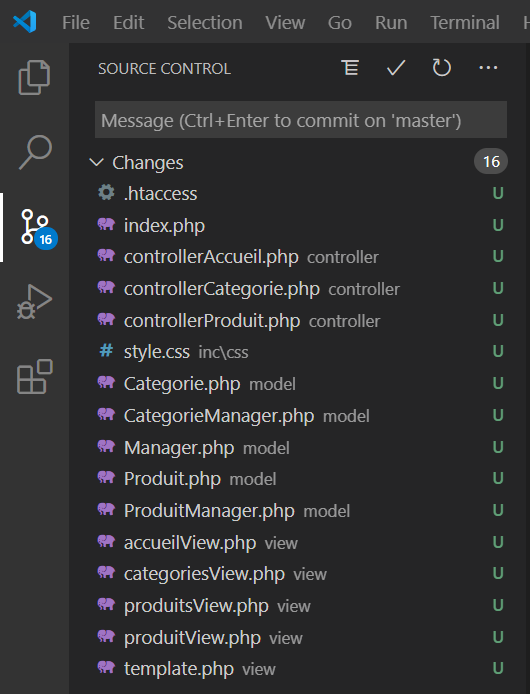
Dans ce laboratoire, vous explorerez les différents outils du système de gestion de version Git.

**Étape préliminaire**

1. Créez un compte sur Github.
2. Installer et configurer Git (voir note de cours)

**Partie 1 – Intégrer votre projet MVC à GitHub**



* Dans Visual Code, avec votre projet ouvert, cliquez sur l’icône de gestion de version se situant dans la barre latérale à gauche.
* Appuyez sur Publish to GitHub puis accepter d’entrer vos informations de connexion GitHub.
* Une page Web apparaitra vous demandant d’accepter le lien entre VS Code et GitHub. Suivez les indications.
* Un popup vous demandera d’ouvrir un lien dans VS Code, Acceptez.
* VS Code vous demandera alors si vous souhaitez publier le projet sur GitHub de manière public ou privé. Sélectionnez Private. 
* On vous demande ensuite les fichiers à gérer, Par défaut, tout est sélectionné. Appuyez sur OK.
* Vous devriez maintenant voir ceci à gauche :  
  Le « U » en vert signifie que ce fichier est à jour.  
  Quelques instants plus tard, il deviendront à « A » pour Added.  
  Ce qui signifie qu’il faut les soumettre pour les ajouter.
* Pour soumettre les fichier, cliquez sur le crochet.  
  Ceci fera un « commit ». Vous devrez alors entrer un message  
  décrivant ce commit. Écrivez : « Dépôt initial »
* Nous avons fait le premier commit, mais le tout est demeuré local pour le moment. Il faut donc pousser le tout sur GitHub.
  + Dans le navigateur, allez sur votre compte GitHub. Vous devriez trouver votre dépôt dans la liste des repositories.
  + Sélectionner-le et copier l’URL qui est indiqué dans la section Quick Setup.
  + De retour dans VS Code, appuyer sur les « … » à côté du crochet et sélectionnez « Push ». On vous indiquera que le dépôt n’a pas de connexion configurée. Appuyez sur « Add remote ».
  + Une boite s’affiche dans le haut. Coller l’URL que vous avez copier sur GitHub. On vous demandera un nom pour cette connexion. Vous pouvez simplement taper « GitHub ».
  + Une fenêtre de login s’affiche, utiliser le moyen de connexion qui vous convient Web ou personnal Token. Puis autoriser « Git Authentification Manager ».
  + De retour dans VS Code, appuyez sur les « … » à côté du crochet et sélectionnez « Pull, Push » puis « Push to ». Une boite s’ouvrira en haut pour vous laisser sélectionner où vous souhaitez pousser vos modifications. Sélectionner votre dépôt sur GitHub.
  + Afin de valider que vos fichiers sont bien rendus sur GitHub, allez sur votre compte GitHub dans le dépôt et vous devriez y trouver la liste des fichiers.

Une image contenant texte, capture d’écran, intérieur, plusieurs

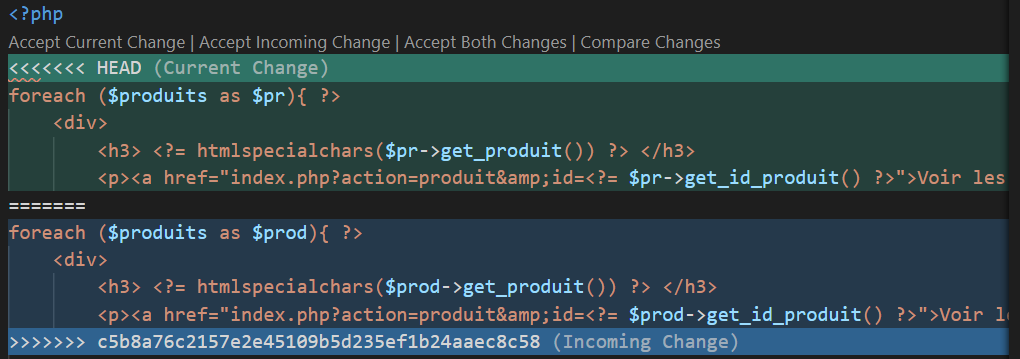
Description générée automatiquementColler ici une capture d’écran de GitHub montrant que les fichiers sont bien présents

**Partie 2 – Gérer le dépôt**

Testons quelques commandes Git et simulons que nous travaillons à plusieurs.

* Créer un dossier sur votre bureau nommé « Collègue imaginaire ».
* Ouvrir un terminal et déplacez-vous dans ce dossier.
* Cloner votre dépôt de GitHub dans ce dossier avec ces commandes suivantes :

Pour obtenir l’URL de votre dépôt, aller sur votre dépôt GitHub cliquez sur le bouton Code.

* + git clone *adresse\_URL\_du\_dépôt*
  + git config --global user.name ‘Votre nom’
  + git config --global user.email votre@courriel.com
* Ouvrez ce dossier dans VS Code et renommez la variable $produit par  
  $prod dans le fichier view/accueilView.php. Enregistrer le fichier.
* Ouvrez le dossier d’origine qui se trouve dans htdocs avec VS Code et renommez la variable $produit par  
  $pr dans le fichier view/accueilView.php. Enregistrer le fichier.
* Dans le projet sur le bureau, faites un commit et un push.
* Dans le projet sur le htdocs, faites un commit et un push.
  + Un message d’erreur devrait s’afficher, car une nouvelle version du code est disponible.
  + Effectuer un « Pull ». Un conflit sera détecté.   
      
    Des boutons au-dessus des conflits vous permettes de gérer le conflit.   
    « Current Change » représente votre version et « Incoming change » représente les changements distants.  
      
    Sélectionnez « Accept Incoming change. »  
      
      
      
    Effectuez ensuite un Commit et un Push. Dans ce cas un message préformaté est indiqué pour le merge.

Une image contenant texte

Description générée automatiquementColler ici une capture d’écran de GitHub montrant que les fichiers sont bien présents